Managing the Domain/Access Application

Order No. 010173
Revision 01

Apollo Computer Inc.
330 Billerica Road
Chelmsford, MA 01824

Chapter 1

Planning for the Domain/Access Application

This chapter introduces you to the Domain/Access product and describes its software, hardware, and telecommunications requirements. Appendix A summarizes all of these requirements in a convenient checklist. Tear out the planning checklist and use it with this chapter as you plan for the Domain/Access application.

Introduction

The Domain/Access product is an application that allows users to transparently access files on Digital Equipment Corporation (DEC*) VAX/VMS computers from their Apollo® workstations. Users can perform most of the same operations on VAX/VMS files that they can on Domain files, so the VAX computer appears to be part of the Apollo network. Specifically, with the Domain/Access application,you can use the following commands and utilities on VAX/VMS directories and files:

- Most Aegis™ and UNIX* shell commands, such as the Aegis **wd** (working directory) command and the UNIX **cd** (change working directory) command
- Language-level input/output procedures, such as the Pascal **open** and **read** procedures
- Certain Aegis system calls, such as **ios_$** calls
- Certain Domain/IX system calls

For a complete list of the commands, procedures, and system calls supported by the Domain/Access product, refer to *Using the Domain/Access Application*.

The Domain/Access application supports operations on all VMS file types, except for VMS files types that don't have an equivalent Domain file type. (For example, you may copy VMS indexed files, which have a .idx extension, to your Apollo workstation, but you can't access them as indexed files because Aegis doesn't have an equivalent file type.)

Understanding Domain/Access Architecture

To help you plan for, configure, and manage the Domain/Access application, we want to explain its architecture and how that architecture affects the way the application operates.

* DEC is a registered trademark of Digital Equipment Corporation. UNIX is a registered trademark of AT&T in the USA and other countries.

The Domain/Access application depends on the Transmission Control Protocol/Internet Protocol (TCP/IP) to communicate with the VAX computer. Thus, the VAX computer needs to be running TCP/IP software. Figure 1-1 shows this relationship.

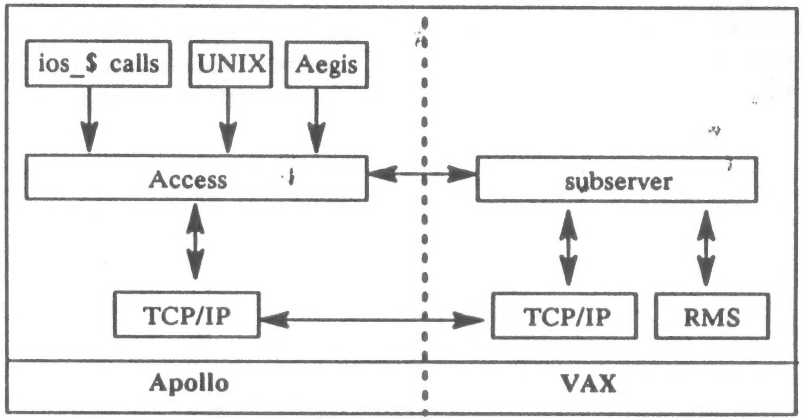

Figure 1-1. Relationship of Apollo Workstation and VAX Node

For the VAX, we support TCP/IP software from two different manufacturers — Excelan, Inc. and The Wollongong Group, Inc. In addition, we support controllers from Excelan (to support Excelan software) and from other manufacturers (to support Wollongong software). Figure 1-2 illustrates a sample network configuration for the Domain/Access product.

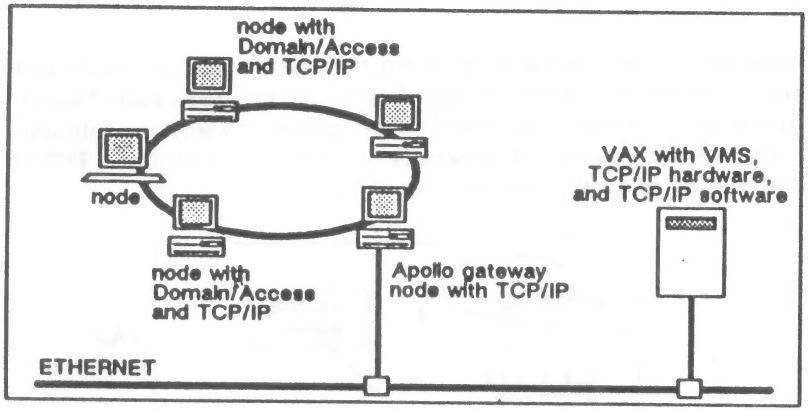

Figure 1–2. A Sample Network Configuration for the Domain/Access Application

The Domain/Access application itself consists of the following components:

- **A type manager** on each user's Apollo workstation that translates the Domain object types and record formats into the appropriate VAX ones, and vice versa
- **A gateway object** on an Apollo workstation that contains the connection information needed by the type managers to establish TCP/IP connections with the VAX and to log in a user
- **A server** on the VAX computer that manages the command and system call requests from Domain/Access users

You copy the server from an Apollo workstation on which you installed the Domain/Access product to the VAX computer you want to access. In addition, you create a VAX gateway for each user. (Chapter 2 explains more about gateway objects.)

When a user includes the name of the gateway object in a pathname, the type manager on their workstation opens the gateway object and uses the information in it to establish a TCP/IP connection with the

server on the VAX, and logs the user in to the VAX. Each time a Domain/Access user is logged in to the VAX, the Domain/Access server spawns a new VAX process, called a **subserver**, to manage the execution of the commands or system call requests for the particular user. (A subserver is spawned for each user/workstation pair. Therefore, if a Domain/Access user is logged in to two different workstations, they'll use two different subservers.) Figure 1–3 illustrates the creation of such subservers.

If a user makes requests that require different Domain processes (such as the Display Manager (DM) process and a shell process), the Domain/Access software creates separate TCP/IP connections *for each process*. However, the same subserver manages these separate TCP/IP connections for each user. Thus, each subserver can support more than one TCP/IP connection, and the number of TCP/IP connections is determined by the number of processes that the user creates on his or her Apollo workstation.

For example, if the first request a user makes to the VAX is the request to read a file (the READ key), the Display Manager (DM) process initiates a TCP/IP connection. So, if the user continued to read and edit a number of different files on the VAX, only one TCP/IP connection to the subserver in the VAX would be required. The subserver requires only one TCP/IP connection because the DM process manages both the edit and read functions.

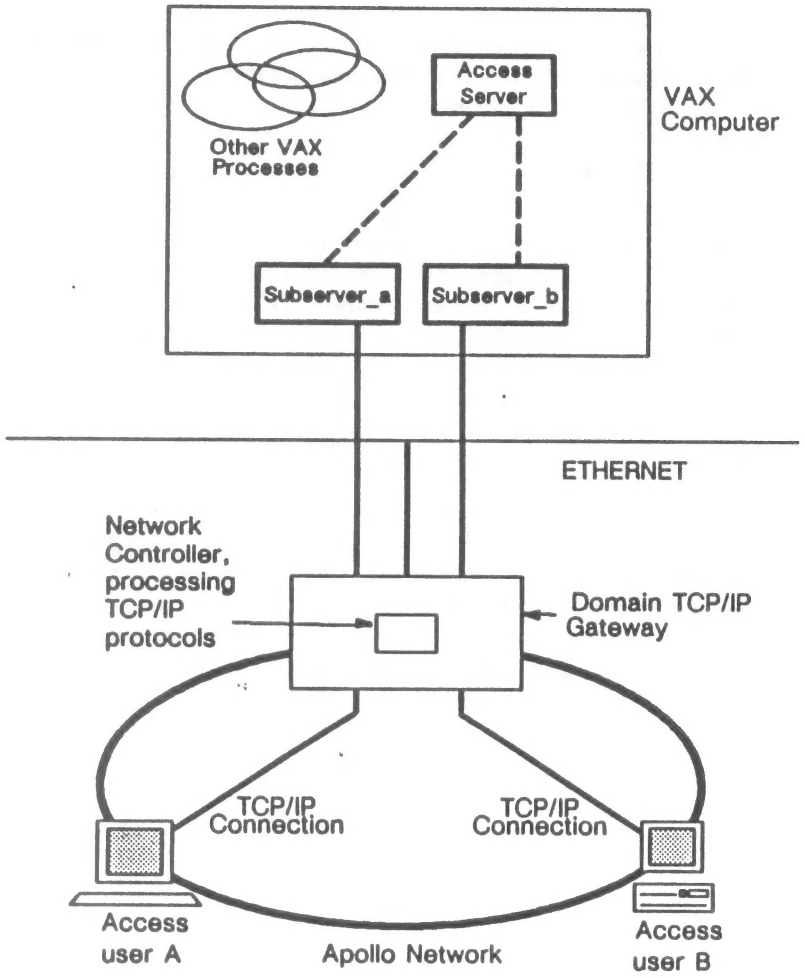

Figure 1–3. Creating Domain/Access Subservers

However, if a user executes different shell commands or system calls from different processes on the Apollo workstation, each Domain process must initiate a separate TCP/IP connection with the subserver in the VAX. Figure 1–4 illustrates the creation of multiple TCP/IP connections and subservers on the VAX.

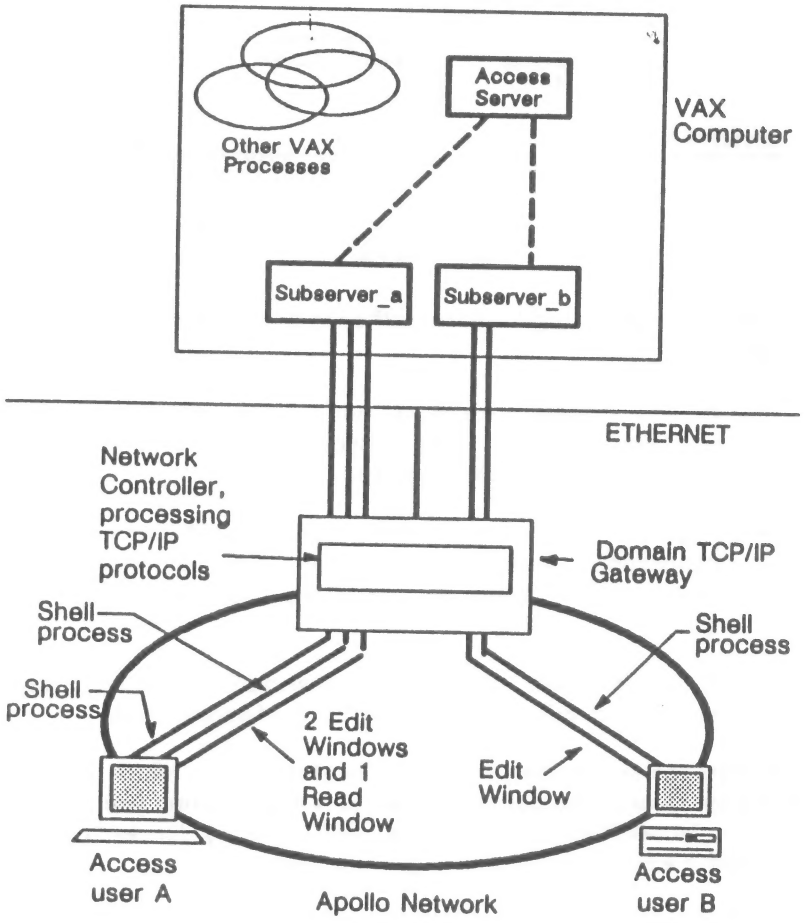

Figure 1–4. Creating Multiple TCP/IP Connections with Domain/Access Subservers

Estimating the Number of Users

From this discussion of Domain/Access architecture, you'll notice that several factors affect the number of people who can use the Domain/Access application simultaneously. Thus, we can't stipulate that the Domain/Access software supports some specific number of users. However, we will explain the factors that affect usage, so that you can estimate the number of users based on your VAX configuration.

The following factors affect how many processes on your VAX will be available for the Domain/Access application:

- The size of your VAX
- The amount of memory installed on your VAX
- The total number of processes available on your VAX (usually, this number is configurable)
- The number of the total processes that are required to run various VAX programs and utilities
- The number of the total processes that users need to do their daily work
- The number of processes for which your VAX TCP/IP hardware and software is licensed

These factors are listed in descending order of importance. Thus, the size of your VAX and the memory installed on it affects the number of users the most; for example, a MicroVAX is much smaller than a VAX 11/780 and probably won't support as many Domain/Access users. The last three factors in the list combine to determine the *load* on your VAX, which also affects the number of Domain/Access users.

For example, suppose you have your VAX (an 11/780 with additional memory installed) configured for a maximum of 32 processes. If the processes required for VAX programs and utilities *and* the processes required by users for normal usage amount to a total of 14

processes, you will have 18 processes (32 total processes minus 14 processes in use) available for the Domain/Access application. However, the availability of these 18 processes doesn't mean that they can all be assigned to Domain/Access users and still maintain a fast response time. Therefore, if your VAX has a moderate load of usage, and several users plan to use the Domain/Access application heavily, you may not be able to add more users and still maintain a reasonable response time.

File Size Limitations

You can only access VAX/VMS files with up to 2,147,483,647 (or 2^{31}) records. Each record can be up to 4094 bytes long.

Planning for Software

You need the following Apollo software installed on a workstation to use the Domain/Access product:

- Aegis *or* Domain/IX operating system, Software Release 9.5 or later
- *One* of the following Apollo TCP/IP products:
  - Domain TCP/IP (version 3.0 or later)
  - Domain/IX BSD4.2 TCP/IP (version 3.0 or later)
- Domain/Access product, release 2.0 or later

Refer to *Planning for TCP/IP* (010483) and to *Planning Domain Networks* (009916) for more information about installing the TCP/IP product on your Domain network.

Note that these Domain software products are listed in the order that they should be installed. You should have an operating system and TCP/IP already running before you use the Domain/Access application.

In addition, you need the following software running on your VAX to use the Domain/Access application. The type of VAX computer you have determines the choices you'll make.

- VMS operating system, Version 4.2 or later *or* MicroVMS operating system, Version 4.2 or later
- *One* of the following TCP/IP software packages:
  - EXOS* 8043 (version 3.2 or 3.3) TCP/IP Protocol Package for VAX/VMS Systems, by Excelan, Inc.
  - EXOS 8044-01 (version 3.2 or 3.3) TCP/IP Protocol Package for MicroVAX II Systems, by Excelan, Inc.
  - WIN**/TCP (version 3.0for VMS), by The Wollongong Group, Inc.
  - WIN/VX (version 2.3 for VMS), by The Wollongong Group, Inc.
  - WIN/μVX (version 2.3 for MicroVMS), by The Wollongong Group, Inc.

NOTE: Domain/Access does not provide communication to UNIX operating systems on VAX computers.

Planning for Hardware

The Domain/Access software doesn't require any special Domain hardware, but establishing a TCP/IP connection may require special hardware. For more information about the hardware requirements for TCP/IP, refer to the TCP/IP planning documentation.

Domain/Access runs on the full family of VAX computers.

Establishing a TCP/IP connection to the VAX may require special hardware for the VAX. Refer to the Excelan or Wollongong documentation for a complete list of supported hardware configurations.

* EXOS is a trademark of Excelan, Inc.

** WIN is a trademark of the Wollongong Group, Inc.

Chapter 2

Configuring the VAX Computer and the Apollo Workstation

This chapter describes how to configure the VAX computer and the Apollo workstation for the Domain/Access product. It includes information about copying some Domain/Access software to the VAX, preparing the VAX for Domain/Access users, starting the Domain/Access server on the VAX, and creating and managing gateway objects.

Installing Software on the VAX

Before you begin to configure the VAX, you must have a working TCP/IP connection between the Apollo workstation and the VAX.

You must also have the Access kit installed on your Apollo workstation. Installing the Access kit is described in the Domain/Access Application Release Notes.

In addition, if you're installing TCP/IP for the first time on the VAX or on the Apollo network, you should verify that you can communicate between the VAX and the Apollo network with TCP/IP. To verify this communication, try the **ftp** and **telnet** utilities, which are part of the TCP/IP product. (Refer to the TCP/IP documentation for more information.)

Now, you must copy certain pieces of the Domain/Access software over to the VAX computer *that is the TCP/IP host*. To do this, complete the following steps. (You need the right to write into the **sys$manager** and the **sys$system** files on the VAX to use this procedure, so we recommend that you log in to the VAX as a system manager.)

1. Start the **ftp** utility in a shell process window, and log in to the VAX. For example,

   in UNIX ftp

   ```
   ftp bacchus
   Connected to bacchus
   220 bacchus TCP FTP Service (Version 4.0 at Tue
   Aug 25 15:21-EDT
   Name (bacchus:name_x):system
   Password (bacchus:system)
   331 Enter PASS Command
   230 User logged in, default directory SYS$SYSROOT:
   [sysmgr]
   ```

   where **bacchus** is the Internet name of the VAX node that is the TCP/IP host.

   or in Aegis ftp

   ```
   ftp bacchus
   TCP trying 192.9.9.2
   Connections established.
   >log system syspassword
   331 Enter PASS command
   230 User logged in, default directory sys$sysroot:
   [sysmgr]
   ```

   where **bacchus** is the Internet name of the VAX node that is the TCP/IP host.

2. Set your working directory to **/sys/access/vms**. For example,

   in UNIX ftp

   ```
   lcd //dionysus/sys/access/vms
   ```

   or in Aegis ftp

   ```
   wd //dionysus/sys/access/vms
   ```

   where **//dionysus** is the name of an Apollo workstation that has the Domain/Access software installed on it.

3. Copy the access.com file and the v1ssr.com file to the VAX/VMS **sys$manager** directory.

   in UNIX ftp

   ```
   >cd sys$sysroot:[sysmgr]
   200 Working directory changed

   >send access.com access.com
   200 PORT Command Ok.
   125 File transfer started correctly
   226 File transfer completed ok
   2070 bytes sent in 0.10 seconds (20 Kbytes/s)

   >send v1ssr.com v1ssr.com
   200 PORT command Ok.
   125 File transfer started correctly
   226 File transfer completed ok
   3596 bytes sent in 0.14 seconds (25 Kbytes/s)
   ```

or in Aegis ftp
>cwd sys$sysroot:[sysmgr]
250 cwd command okay.

>send access.com access.com
200 ALLO command okay.
200 PORT command okay.
200 opening data connection for access.com
(197.6.3.15, 1518).
226 Transfer complete.

>send v1ssr.com v1ssr.com
200 ALLO command okay.
200 PORT command okay.
200 opening data connection for v1ssr.com
(197.6.3.15, 1518).
226 Transfer complete.

4. Copy the access.hlp file to the VAX/VMS **sys$help** directory.

in UNIX ftp
>cd sys$sysroot:[syshlp]
200 Working directory changed

>send access.hlp access.hlp
200 PORT Command Ok.
125 File transfer started correctly
226 File transfer completed ok
3209 bytes sent in 0.09 seconds (33 Kbytes/s)

or in Aegis ftp
>cwd sys$sysroot:[syshlp]
250 CWD command okay.

>send access.hlp access.hlp
200 ALLO command okay.
200 PORT command okay.
200 opening data connection for access.hlp

(197.6.3.15, 1518).
226 Transfer complete.

5. Copy the v1access.exe, v1server.exe, and v1subserv.exe files to the VAX/VMS **sys$system** directory. You must send these files as a data type of *image*. Use the **ftp type** command to do this. (The **ftp type** command supports four data types: ascii, binary, image, and local. The image data type specifies a fast transfer of ASCII or binary files.)

in UNIX ftp
>cd sys$sysroot:[syexe]
200 Working directory changed

>type image
200 Command OK

>send v1access.exe v1access.exe
200 PORT Command Ok.
125 File transfer started correctly
226 File transfer completed ok
55296 bytes sent in 1.58 seconds (34 Kbytes/s)

>send v1server.exe v1server.exe
200 PORT Command Ok.
125 File transfer started correctly
226 File transfer completed ok
57344 bytes sent in 1.35 seconds (41 Kbytes/s)

>send v1subserv.exe v1subserv.exe
200 PORT Command Ok.
125 File transfer started correctly
226 File transfer completed ok
77824 bytes sent in 1.86 seconds (41 Kbytes/s)

>quit

or in Aegis ftp
cwd sys$sysroot:[syexe]
250 CWD command okay.

```
>type image
200 Type set to I.

>send v1access.exe v1access.exe
200 ALLO command okay.
200 PORT command okay.
150 opening data connection for access.exe
(197.6.3.15, 1518).
226 Transfer complete.

>send v1server.exe v1server.exe
200 ALLO command okay.
200 PORT command okay.
150 opening data connection for v1server.exe
(197.6.3.15, 1518).
226 Transfer complete.

>send v1subserv.exe v1subserv.exe
200 ALLO command okay.
200 PORT command okay.
150 opening data connection for v1subserv.exe
(197.6.3.15, 1518).
226 Transfer complete.

>quit
```

Preparing the VAX

The VAX system manager at your site must complete the following steps to prepare the VAX that is the TCP/IP host for the Domain/Access product.

1. Verify that all user accounts on the VAX contain the privileges necessary to use Domain/Access. The VMS command **show process/privilege** will show a process's current privileges. The VMS **authorize** utility can be used to check someone else's privileges.

   - For users who need to start the Access server on the VAX as well as access VAX/VMS files, these privileges are **detach, setprv, tmpmbx, world, netmbx, sysprv,** and **sysnam.**
   - For general users who need to access VAX/VMS files, the only privilege necessary is **tmpmbx.**
   - For general users who need to use DECnet to access files on a VAX node other than the TCP/IP host, these privileges are **tmpmbx** and **netmbx.** (Note that users need these privileges on every VAX node that they access through DECnet.)

2. Verify that all users have the appropriate access rights to certain Domain/Access files on the VAX. Use the VMS command **directory/protection** [filename] to check the protection of a file.

   - For users who will need to start the Access server on the VAX as well as access VAX/VMS files, (r)ead and (e)xecute rights are required on the following files:

     sys$manager:access.com
     sys$manager:v1ssr.com
     sys$system:v1access.exe
     sys$system:v1server.exe
     sys$system:v1subserv.exe

   - For general users who will need to access VAX/VMS files, the (e)xecute right is required on the following file:

     sys$system:v1subserv.exe

3. Check that each user's account has the following minimum values for the **astlm** (asynchronous system trap limit), **tqelm** (timer queue limit), and **bytlm** (byte limit) quotas:

   astlm 10
   tqelm 10
   bytlm 8192

Use the VMS command **show process/quota** to show a process's current quotas. Use the VMS **authorize** utility to check someone else's quotas.

(Each VAX user's account has a list of different quotas and the minimum values for those quotas. The quotas determine the system's resources, such as memory, that are allocated for each user.)

4. Remind Domain/Access users that the user ID, password, and account that they put in their gateway must match what you assign for them on the TCP/IP host VAX. If users' passwords are changed their gateways must be updated to reflect the change.

Installing VAX/VMS Help Files

Log in to the VAX/VMS system and execute the command:

```
$ library sys$help:helplib.hlb sys$help:access.hlp
```

This makes Access help available from the VMS help facility. Thus, the command

```
$ help access
```

gives VMS help for Access.

Starting the Domain/Access Server

To start the Domain/Access server, you use the @access command file on the VAX. You start the server on *the VAX that is the TCP/IP host*. You may *not* start multiple servers for the same port or start a server on another VAX node.

Using the @access Command

Before you start the server, you should be familiar with the syntax of the @access command file. @access has the following syntax:

@access *[port] [directory] [debug_level] [process_name] [tcp_type]*

The port Argument

The **port** argument specifies the TCP/IP port number that the Access server will use. The default value for this argument is 1024. (Port numbers 1 through 1023 are assigned or reserved as part of the DARPA Internet protocol. For example, telnet uses TCP port 23.) TCP ports 1024 through 32727 have no specific assignments. As a result, different software manufacturers may use the same port number for different TCP/IP–based products. Thus, you need to specify the TCP port *only* if another product running on an Apollo workstation or the VAX/VMS system requires port 1024.

The directory Argument

The **directory** argument indicates the VAX/VMS directory where the Access server will reside. The default directory for this argument is the **sys$system** directory. (Thus, the directory you choose should match the directory where you copied the software to the VAX in Step 5 of the "Installing Software" section.)

The debug_level Argument

The **debug_level** argument selects the status messages to be displayed in your telnet window (with the exception of * errors, which are displayed on the VAX console). If you specify *, @access starts the server and returns you to the prompt. Specifying any other value causes your process to enter into a "Listen" mode. In Listen mode the process waits for errors and displays them. To leave Listen mode, type ^C at your telnet window. Typing ^C at the telnet window causes the server and its subservers to abort.

Specify * (default) for fatal errors displayed on the VAX console, **f** for fatal errors displayed in your telnet window, **p** for presentation layer messages, **y** for application layer messages, **n** for presentation layer and application layer messages (**p** and **y**), **d** for data dumps, **i** for Record Management Services (RMS) error messages, or **0** (zero) for no messages. (Refer to Chapter 3 for more information about the different debug levels and the messages associated with them.)

The process_name Argument

The process_name argument specifies the name of the process for the Access server on the VAX. The default process name is v1_*port*, where *port* is the TCP port number of the Access server. After the server starts, use the VMS command show process to verify that the server process_name is running.

The tcp_type Argument

The tcp_type argument specifies the type of TCP/IP software on the VAX. Use e for Excelan software (default) or w for Wollongong software.

Procedure to Start the Domain/Access Server

Now that you are familiar with the optional arguments for the @access command file, follow these steps to start the Domain/Access server. (Some commands in this procedure are presented on two lines to accommodate the margins of this manual. Type the command on one line.)

1. Log in as a system manager to the VAX that is the TCP/IP host. For example,

```
telnet bacchus
Trying...
Username:  system
Password:  (not displayed)

          Welcome to VAX/VMS Version 4.2
                    .
                    .
```

   where bacchus is the name of the VAX node that is the TCP/IP host.

2. Ensure that the Domain/Access server starts whenever the VAX is rebooted. To do this, edit the site-specific startup file on the VAX (typically the sys$manager:systartup.com file) to include the following line:


```
@sys$manager:access [port] [directory] [debug_level]
[process_name] [tcp_type]
```

   Note that this line must follow the command that starts TCP/IP. For example,

```
@exos$etc:netstart
@sys$manager:access 1024 sys$system: *
access_server e
```

   or

```
@twg$misc:startinet
@sys$manager:access * * * * w
```

   For some configurations, you may need to add another excelan command to this file to establish two-way communication with some Apollo nodes. Add this command after the @starttcp command. For example,

```
@exos$etc:netstart
route add  node_be99  node_gate
@sys$manager:access
```

   where *node_be99* is the name of the Apollo host node and node_gate is the name of the gateway node.

3. Start the Domain/Access server without rebooting the VAX. (In Step 2, you edited a file that brings up the Domain/Access server each time the VAX is rebooted.) Enter the following command:

```
@sys$manager:access 1024 sys$system: *
 access_server e
```

4. Keep your telnet window open so that you can receive status messages from the VAX. (If you set the debug level of the Domain/Access server to "*", the default, you receive these status messages on the VAX console, not in your telnet window.)

Managing VMS Gateway Objects on the Apollo Workstation

You manage VMS gateways by using the **crvg**, **dlvg** and **svg** shell commands, which are supplied with the Domain/Access application.

Creating VMS Gateway Objects

Now that you have the Domain/Access server running on the VAX, you must provide a way for users to access that server. To do this, you create a *VMS gateway object* on the Apollo workstation, which has an object type of **vl_gate**. The gateway object contains information that a Domain/Access type manager needs to establish a TCP/IP connection with the VAX.

When a user specifies a file on a VAX, he or she enters the name of the gateway object for the VAX, followed by the name of the file to be accessed. For example,

in an Aegis shell
ld *//node/gateway_object*/**dmo:/my.dir**
or in a UNIX shell
ls *//node/gateway_object*/**dmo:/my.dir**

where *//node/gateway_object* is the name of the VMS gateway object and **/dmo:/my.dir** is the name of the directory on the VAX. (Refer to *Using the Domain/Access Application* for details about specifying VMS pathnames.)

When a user puts the gateway object name in a pathname, the type manager recognizes that the name refers to a **vl_gate** object and therefore knows that the part of the pathname to the *right* of the gateway object is for a file or directory on the VAX. The type manager uses the information in the gateway object to log the user in to the VAX, and then accesses the file or directory specified.

NOTE: Provide each user with their own gateway for each VAX. Since the user's login information is part of the gateway object, you should acl the gateway object so that only the proper user has access to it.

Users can access files on other VAX nodes that are connected to the TCP/IP host through a DECnet network. For example, Figure 2-1 shows a configuration for Domain/Access in a DECnet network.

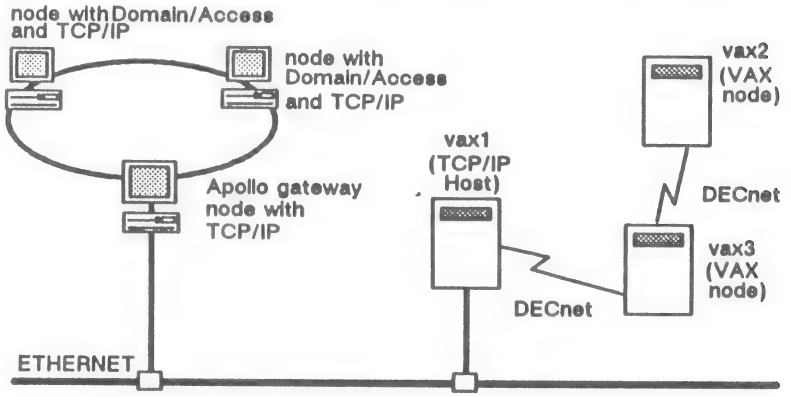

Figure 2-1. Accessing VAX Files through DECnet

In this situation, you could create separate gateway objects that contain the connection information for *vax1*, *vax2*, and *vax3*. Note that the TCP/IP host is the *only* VAX node that *must* run the Access server and *must* have a gateway object associated with it. Users can still access DECnet nodes without separate gateway objects; they simply include the name of the DECnet node in their VMS pathname. Therefore, by creating separate gateway objects, you don't allow users to access these DECnet nodes, you just make it *more convenient* for them to do so.

To create a gateway object, which is a Domain file system object of type **vl_gate**, you use the **crvg** command (in either an Aegis or a UNIX shell). The **crvg** command has the following syntax:

crvg *gate_name host_name user_name password [vax_node] [-p port_no] [-t] [-a account]*

The crvg command takes four to eight arguments in the following order:

1. gate_name, the name of the VMS gateway object (required)
2. host_name, the Internet name of the VAX node that is the TCP/IP host (required)
3. user_name, the user name for the VAX(required)
4. password, the user password for the VAX(required)
5. vax_node, the name of the VAX node to be accessed via DECnet (optional)
6. -p followed by a TCP/IP port number greater than 1023 (optional)
7. -r the replace option (optional)
8. -a followed by an account name (optional)

The gate_name Argument

This argument specifies the name of the gateway object that crvg will create. When the crvg command creates the gateway object, it places the Internet name, VAX node name,VAX user name, VAX password, VAX account (if there is one), and the port number in the object.

The host_name Argument

The host_name argument specifies the name of the VAX node that is the TCP/IP host. This argument is case-sensitive.

This argument must match the entry for this TCP/IP host in the Domain TCP/IP host tables.

The vax_node Argument *(optional)*

The VAX/VMS node name (also known as the DECnet name) identifies the VAX/VMS system that contains the files you need. If you

specify this argument, the file is accessed using DECnet form the VAX. Do not use this argument if you want to access files on the VAX node that is the TCP/IP host.

This argument must match the DECnet name of the VAX node where the files are to be accessed.

The user_name Argument

The user_name argument specifies the name to use for logging in to the VAX. This argument is case-insensitive.

The password Argument

The password argument specifies the password to use for logging in to the VAX. This argument is case-insensitive.

The [-p] Option

The TCP port number identifies the endpoints of a particular TCP/IP connection. The port number in *all* gateway objects *must* match the port number that you assigned when you started the Domain/Access server with the @access command. (See the "Starting the Domain/Access Server" section, earlier in this chapter, for more information.)

Thus, the default value for the [-p] option is the same (1024) as the default value for the [port] argument of the @access server. Port numbers 1 through 1023 are assigned or reserved as part of the DARPA Internet protocol. For example, telnet uses TCP port 23. However, TCP ports 1024 through 32727 have no specific assignments. As a result, different software manufacturers may use the same port number for different TCP/IP-based products. You should choose a port number that doesn't conflict with any other existing products in your network. Thus, you need to specify the TCP port *only* if another product running on an Apollo workstation or the VAX/VMS system requires port 1024.

The [-r] Option

The [-r] option replaces any existing file system objects that have the same name as the gateway object you create. For example, when you

specify this option, an ASCII file (type *uasc*) with the same name as the VMS gateway object would be replaced by the VMS gateway object.

The [-a] Option

The [-a] option allows you to specify a VAX account name for the login sequence. This is not required and is used for accounting purposes on the VAX.

Example

For example, suppose you create a gateway object for every VAX node in Figure 2-1. You'd use the following commands:

```
crvg //node_x/v_vax1 vax1 smith password ::
crvg //node_x/v_vax2 vax1 smith password vax2::
crvg //node_x/v_vax3 vax1 smith password vax3::
```

where the @access server is running on port 1497 of vax1.

Deleting VMS Gateway Objects

The **dlvg** (delete VMS gateway object) command deletes VMS gateway objects (in either an Aegis or UNIX shell). It removes the associations established by a **crvg** command. After you execute the command, you cannot use the deleted gateway object to access VMS files.

The **dlvg** command has the following syntax:

dlvg *gate_name*

where the gate_name argument specifies the name of the gateway object. You can use wildcards in the **dlvg** command. When you use wildcards, **dlvg** requests a yes or no response before you delete each matching VMS gateway. For example, you can issue the following command to delete gateways in your current working directory.

```
dlvg ?*
delete "v_london" ? n
delete "v_paris" ? y
```


Showing VMS Gateway Objects

Use the **svg** (show VMS gateway object) command (in either an Aegis or UNIX shell) has the following syntax:

svg [*gateway_name*]

The **svg** (with no arguments) command displays the following information about the VMS gateway object:

- Internet host name
- TCP port number
- VAX/VMS DECnet node ID
- User Name
- Password
- Account

The **svg** command displays this information for every gateway object in your current directory. To specify a particular gateway object, use the optional **gateway_name** argument. You can use wildcards with the **gateway_name** argument to display information about several VMS gateways. For example:

```
svg //node_b/v_?*

gateway name       tcp host name    user_name    password
//node_b/v_paris   vax1             smith        wingo
//node_b/v_london  vax1             smith        namu
//node_b/v_vax3    vax3             smith        nafriel

account          port          gateway point to
research         1024          paris::
research         1024          london::
research         1024          ::
```

Chapter 3

Isolating Problems

This chapter explains how to isolate problems with the Domain/Access software by interpreting the error messages from the Domain/Access server on the VAX.

In Chapter 2, you learned how to use the @access command to start a Domain/Access server on the VAX computer. The @access command has five arguments, as follows:

@access *[port] [directory] [debug_level] [process_name] [tcp_type]*

When you start a Domain/Access server for debugging, you keep your telnet window open so that you can receive messages about the status of the server. By varying the value of the **[debug_level]** argument, you receive different messages from the server, which may help you isolate problems with the Domain/Access software. (Varying the value of the other arguments doesn't help isolate software problems.)

Changing Debug Levels

The **[debug_level]** argument determines what types of error messages appear in the telnet window at your workstation. (The only exception is the * debug level, which displays the errors on the VAX console, a terminal that is connected directly to the VAX, instead of in your telnet window.) You may select from the following debug levels:

- ***** (Default) displays fatal errors (from both the presentation and application layers) on the VAX console
- **f** Displays fatal errors (from both the presentation and application layers) in your telnet window
- **p** Displays status messages in the presentation layer of the software in your telnet window
- **y** Displays status messages from the application layer of the software in your telnet window
- **n** Displays status messages from the presentation and application layers of the software (combination of debug levels p and y (data dump)in your telnet window)
- **d** Displays the data written to or read from the VAX *and* the application layer messages (y) in your telnet window
- **l** Displays the Record Management Services (RMS) errors that correspond to different Domain errors in your telnet window
- **0** Displays no messages or errors

You can run the Domain/Access server with one debug level at a time. Therefore, in isolating a problem, it may be necessary to stop the @access server, reset the debug level, and start the server again. You can repeat this process with different debug levels until you achieve the detail you want.

Interpreting Fatal Errors

When you run a server in the * debug mode, which is the default debug mode, or in the **f** debug mode, you receive only the fatal errors — the errors that may cause Domain/Access to stop unexpectedly. These errors usually have the following format:

```
MODULE:  MESSAGE
```

For example, you may receive the following fatal error:

```
$P_REC_M_AST:  MEMORY ALLOCATION FAILED
```

To interpret this error, look for the message portion in Table 3–1 and follow the suggested action. If you need to report the error to your customer service representative, make sure you give both the module ($P_REC_M_AST, for example) and the message. (To contact customer services within the United States, call 1–800–227–6556. Outside the U.S., contact your local Apollo office.)

Note: Messages from the Excelan or Wollongong software aren't included in Table 3–1. If you receive an error that begins with "E", such as ENOBUFS, you should look in the Excelan documentation for an explanation of the error. If you receive an error that begins with "WIN", you should look in the Wollongong documentation for an explanation of the error.

Table 3-1. Fatal Errors (Debug Level * or F)

Error	Description/Action
NO PORT AVAILABLE	All of the TCP/IP ports are busy. Try operation again, and if error persists, restart TCP/IP server and the access server on the VAX. If the error persists, report it to your customer service representative.
ERROR IN $READEF ERROR IN $SETEF ERROR IN $CLREF ERROR IN $WFLOR ERROR IN $GETJPI ERROR IN $CANTIM PROC_HANDLE NOT FOUND BUG CHECK - PLEASE REPORT FATAL ERROR	Report this error to your customer service representative.
IGNORING EXTRA ARGUMENT	There is an error in the arguments that you specified when starting the @access server.
INVALID IMAGE	Verify that the .exe files have world privilege. Start the server again.
MISMATCHED VERSION	The versions of **v1access.exe**, **v1server.exe**, and **v1subserv.exe** are incompatible. Install new versions from your latest software kit.
TIMEOUT OCCURRED	The Domain/Access software timed out while waiting for an event.

Table 3-1. Fatal Errors (Debug Level * or F) (Continued)

Error	Description/Action
ERROR IN QI/O(W) SOCKET CREATE ERROR IN QI/O (W) SOCKET ACCEPT	The TCP/IP connection didn't get created correctly. The Domain/Access software may have timed out waiting for a connection. Try again. If the error persists, report it to your customer service representative.
MEMORY ALLOCATION FAILURE MEMORY DEALLOCATION FAILURE	The working set quota, one of the quotas in the user authorization file on the VAX, is too low for the current load on the VAX. See your VAX system manager for help.
ERROR IN ASSIGN MAILBOX ERROR IN CREATE MAILBOX ERROR IN QI/O (W) MAILBOX READ ERROR IN QI/O (W) MAILBOX WRITE ERROR IN ASSIGN SOCKET ERROR IN $GETDVI NO SUCH DEVICE ERROR IN DEASSIGN SOCKET	An error occurred with the Excelan driver, the Wollongong driver, or the mailbox. Verify that the user has both **tmpmbx** and **netmbx** privileges on the VAX. If the user doesn't have these privileges, request them from your system administrator. Check the gateway object for accuracy, and that the TCP/IP is running. If you still can't find an error, reboot the TCP/IP server and the Domain/Access server on the VAX and try the operation again. If the error persists, report it to your customer service representative.

Table 3-1. Fatal Errors (Debug Level * or F) (Continued)

Error	Description/Action
INVALID HEADER LENGTH INVALID MESSAGE LENGTH INVALID MESSAGE TYPE INVALID P_HEADER LENGTH STRING TOO LONG	The data on the TCP/IP link is corrupted. Run the TCP/IP diagnostics. If no problem is found with the TCP/IP link and the problem persists, report the error to your customer service representative.
CONN_HANDLE NOT FOUND ERROR IN DEASSIGN MAILBOX ERROR IN DELETE MAILBOX EMPTY SOCKET SEND QUEUE	Warning only — no action necessary.
INVALID PORT	Try the operation again, and if the problem persists, report the error to your customer service representative.
PID NOT FOUND	The subserver exited unexpectedly. Someone may have stopped the subserver process on the VAX. If the problem persists, report the error to your customer service representative.

Table 3-1. Fatal Errors (Debug Level * or F) (Continued)

Error	Description/Action
ERROR CREATING PROCESS	The Domain/Access software couldn't create a process on the VAX. Check to make sure that the correct executable file is in the correct directory on the VAX. Check to make sure that you have the rights to execute the files. Also, check that the process does not exceed any user or system quotas. Finally, check to see if a process already exists with the same name.
ERROR IN QI/O (W) SOCKET SELECT ERROR IN QI/O (W) SOCKET SEND ERROR IN QI/O (W) SOCKET RECEIVE	A TCP/IP error occurred or a TCP/IP connection disappeared. A ring failure, an ETHERNET failure, or a TCP/IP failure on the gateway node, on the VAX, or on the client node may have caused the problem. Consult your TCP/IP documentation for diagnosing TCP/IP errors.
ERROR IN $CRELNM	An error occurred when entering a logical name in the VAX system tables. Either the logical name already exists, or the access server doesn't have the access privileges to read the VAX system tables. Verify that you have the necessary privileges and restart Access. If the error persists, report it to your customer service representative.
ERROR IN RMS$SETDIR	The access server couldn't set the working directory to the user's directory. Ask your VAX system manager to check the sysuaf.dat file for accuracy

Table 3-1. Fatal Errors (Debug Level * or F) (Concluded)

Error	Description/Action
ERROR IN $SETIMR AST OR TIMER QUOTA EXCEEDED	The ast quota or the timer quota is too low for the current load on the VAX. See your VAX system manager for help.
ERROR IN OPENING FILENAME ERROR IN ACCESSING FILENAME ERROR IN CLOSING FILENAME	The access server doesn't have the access privilege to read specified system file, either the Apollo TCP/IP file **rhost** or the VAX/VMS file **sysuaf.dat**. See your system manager to change this.
INVALID PASSWORD FOR USER INVALID USER	The access server encountered an invalid user ID, password, or account. Make sure that the user ID, password, and account in the user's gateway matches what the VAX system manager has assigned in the VAX, and verify that the password has not changed or expired.

Interpreting Application and Presentation Layer Messages

Both the **p** and **y** debug modes are similar to a trace facility, where you can monitor all of the activity in a particular layer. The **p** and **y** debug modes contain many messages that refer to the internals of the Domain/Access product. Therefore, you should have your customer service representative help you interpret these messages.

For example, if a fatal error (debug level *) occurs, and you don't understand the cause of the error, you should do the following:

- Run a server in **y** mode
- Repeat the operation that caused the fatal error
- Run another server in **p** mode
- Repeat the operation that caused the fatal error

A long series of messages appears in the telnet window where the server was started. You should save the messages that appear up until the error and go over these messages with your Apollo customer service representative.

If receiving this long series of messages in your telnet window isn't convenient, you can run the server in the **y** or **p** debug mode as a batch job on the VAX and save the output log for perusal later. (Refer to the VAX/VMS documentation for information about submitting batch jobs on the VAX.)

Interpreting Data Dumps

In addition to the application layer messages, the **d** debug mode displays the data that is sent to or received from the VAX when a user issues a command or system call. The **d** debug mode illustrates how object types and record formats are transformed between the Domain and VAX file systems.

Interpreting RMS Errors

When you run the Domain/Access server in the default debug mode (*), it returns errors to you in the Aegis format, such as "name not found (OS/naming server)" (for commands) or "ios_$name_not_found" (for system calls). However, some Aegis errors may result from a number of different VAX errors, which are reported by Record Management Services (RMS). Thus, to correct some errors with Domain/Access, you may need to know the RMS error that caused the Domain/Access error.

To find the RMS code responsible for an error, run the Domain/Access server with the **i** debug mode. In this mode, the RMS error codes appear in your telnet window. Table 3-2 lists the Aegis error

codes, the RMS errors that may cause each Aegis error, and the descriptions of each RMS error. For complete error recovery instructions for each RMS error, see DEC's *VAX/VMS Record Management Services Reference Manual.*

Table 3-2. RMS Errors (Debug Level I)

Aegis Error Code	RMS Error Code(s)	Description(s)
ios_$already_exists	RMS$_FEX	File already exists, not superseded.
ios_$bad_file_hdr	RMS$_ATR	Read error on file header.
	RMS$_ATW	Write error on file header.
	RMS$_IFA	Invalid file attributes; file header corrupted.
ios_$bof_error	RMS$_BOF	Beginning of file detected (warning).
ios_$buffer_too_big	RMS$_RTB	Record too large for user's buffer (warning).
	RMS$_USZ	Invalid user record area size.
ios_$cant_initialize	RMS$_CRE	ACP file creation failed.
ios_$concurrency_violation	RMS$_ACT	File activity precludes operation.
	RMS$_FLK	File currently locked by another user.
	RMS$_RLK	Target record currently locked by another stream.

Table 3-2. RMS Errors (Debug Level I) (Continued)

Aegis Error Code	RMS Error Code(s)	Description(s)
ios_$end_of_file	RMS$_EOF	End of file detected.
ios_$illegal_name_redefine	RMS$_IDR	Invalid directory rename operation.
ios_$illegal_operation	RMS$_IOP	Invalid operation attempted.
ios_$insufficent_rights	RMS$_PRV	Insufficient privilege or file protection violation.
ios_$name_not_found	RMS$_DEV	Bad device or inappropriate device type for operation.
	RMS$_DIR	Error in directory name.
	RMS$_DNF	Directory not found.
	RMS$_FNF	File not found.
	RMS$_FNM	Syntax error in file name.
	RMS$_LNE	Logical name error.
	RMS$_NOD	Node name error.
	RMS$_QUO	Error in quoted string.
	RMS$_SYN	Syntax error in file specification.
	RMS$_TYP	Error in file type.
	RMS$_VER	Error in version number.
	RMS$_WCC	Invalid wildcard context number.
	RMS$_WLD	Invalid wildcard operation.

Table 3-2. RMS Errors (Debug Level I) (Continued)

Aegis Error Code	RMS Error Code(s)	Description(s)
ios_$never_closed	RMS$_CCF	For the sys$rmsrundwn routine, an output file could not be closed successfully.
ios_$object_not_found	RMS$_ACC	File access error.
	RMS$_DNR	Device not ready, not mounted, or unavailable.
	RMS$_FND	ACP file or directory lookup failed.
ios_$object_read_only	RMS$_WLK	When access to write was attempted, the device was locked for read only.
ios_$put_bad_rec_len	RMS$_MRS	Invalid value for maximum record size.
	RMS$_RSZ	Invalid record size.
ios_$resource_lock_err	RMS$_DEADLOCK	The VMS lock manager detected a deadlock.
	RMS$_MKD	ACP could not mark file for deletion.
	RMS$_RSA	Record stream currently active (asynchronous operations).
status_$ok	RMS$_NORMAL	Operation successful.
access_$insufficient_memory	RMS$_DME	Dynamic memory exhausted.

Table 3-2. RMS Errors (Debug Level I) (Concluded)

Aegis Error Code	RMS Error Code(s)	Description(s)
access_$no_such_program	RMS$_NET	Network operation failed at remote node.
	RMS$_NETFAIL	Network operation failed at remote node.
access_$vs_fatal_error	RMS$_CHN	Channel assignment failure.
	RMS$_CRC	Network DAP level CRC check failed on Close.
	RMS$_FUL	Device full.
	RMS$_PENDING	An asynchronous operation was initiated but is not yet completed.
	RMS$_PLG	Error in file prolog.
	RMS$_TMO	Timeout period expired.
	RMS$_UBF	Invalid user record area address.
	RMS$_WER	File write error.

Using the @v1ssr Command File

The @v1_ssr command file is an interactive version of the @access command file. Thus, @v1_ssr takes the same arguments as @access, but it will prompt you for those arguments (so you don't have to remember them). When you type @v1_ssr in your telnet window, you'll see the following prompt:

```
Enter the pathname of parameter file:
```

Since parameter files are unsupported, enter <RETURN> for the next prompt, which looks like the following:

```
Enter command (start/stop/help) [default=start]:
```

Start is the only command supported. Next, you receive the following prompt:

```
Enter TCP/IP type (Wollongong or Excelan) [default=E]:
```

Once you've selected a type of TCP/IP software, the command file prompts you for a TCP/IP port, as follows:

```
What port will be used for initial connections [default =
1024] ?
```

After you enter the port number, the following prompt appears:

```
What level of debug messaging do you want – *(default=fa-
tal), F, P, Y, D, N, I(informational), or 0(zero)?
```

Once you supply the debug level, the command prompts you for the process name as follows:

```
What do you wish the process name of the server to be [de-
fault = v1_port] ?
```

Finally, after you select a process name, the final prompt appears:

```
What directory do the server and subserver reside in [default
= sys$system] ?
```

After you enter the process name of the server, a Domain/Access server is created on the VAX with the characteristics you specified at the prompts.

Chapter 4

Using the access_diag Diagnostic

This chapter explains how to use the Domain/Access diagnostic tool, **access_diag**, which is run on an Apollo workstation and which provides a loopback diagnostic test. This test sends a test message to each of the following software components. Then, the software being tested returns the message.

- Local Domain/Access type manager
- Local Access transport layer
- Local TCP/IP

- Remote Access transport layer
- Remote Domain/Access (This test performs a login, tests sending and receiving messages from the VAX, and then logs out.)

Starting access_diag

You start the Domain/Access diagnostic tool by entering the following command at the prompt:

```
/sys/access/diagnostics/access_diag
```

The diagnostic screen, shown in Figure 4–1, appears. This screen consists of a menu in the top third of the window and the shell transcript pad in the bottom two thirds of the window. You use the menu to select the tests. The results appear in the transcript pad and remain there after you exit the diagnostic.

If the window is not large enough, some parts of the menu may be *clipped* and therefore cannot be selected. Clipped windows have a gray bar on the right or on the bottom. To avoid clipped windows, simply grow the window so that you can see the whole menu.

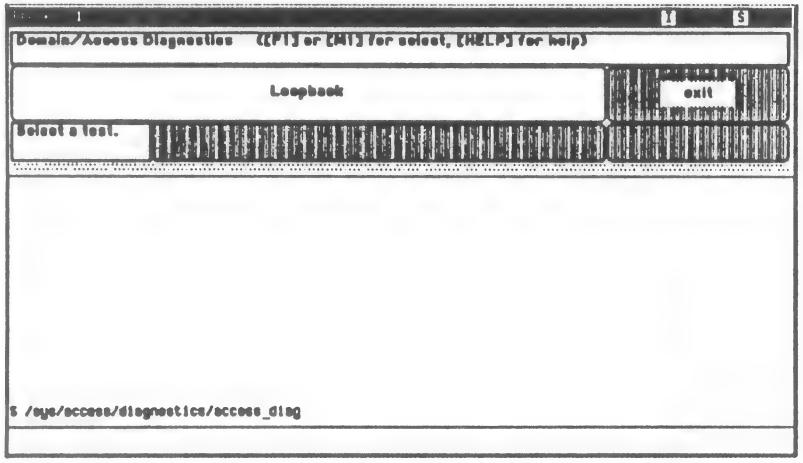

Figure 4–1. The access_diag Menu

Completing the Loopback Test

To complete the loopback test from the menu, follow these steps.

1. Select "Loopback," as shown in Figure 4–1. To do this, complete the following steps.

   A. Move the locator cursor, an arrow, with the touchpad, mouse, or cursor movement keys over the "Loopback" selection. The menu highlights the selection.

   B. Press the M1 (left) mouse key or the F1 function key. The bottom section of the menu changes to show the selections available for the loopback test, as shown in Figure 4–2.

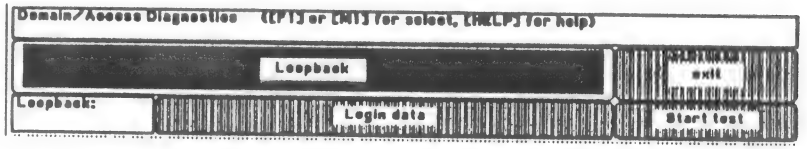

Figure 4–2. The Loopback Test Menu

2. Select "Login data."

   A pop–up menu appears with four fields for you to enter the login data. Figure 4–3 shows the login data pop–up menu.

3. Enter your User ID, your password (if any), and the TCP/IP host name. You can also specify the host TCP/IP port if you are not using the default (port 1024). This login data remains in effect until you either exit the diagnostic or enter new values.

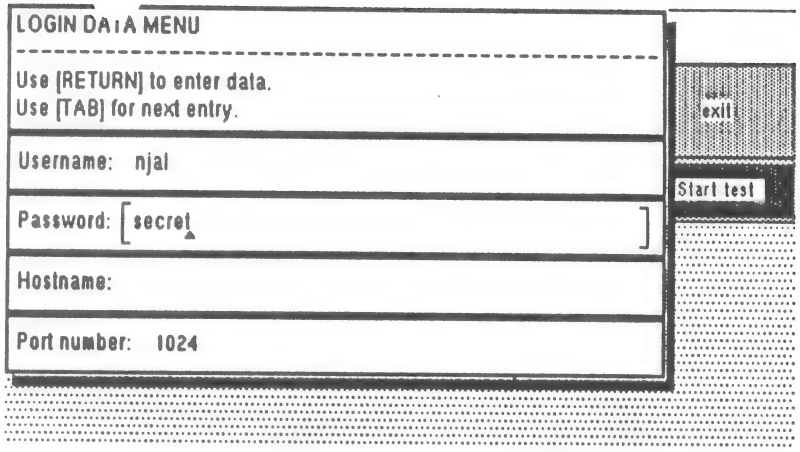

Figure 4–3. The Login Data Pop–up Menu

To enter values in each field of the pop–up menu, follow these steps.

A. Select the field. The menu displays the typing cursor, a small triangle.

B. Enter the information. When you begin typing, the menu displays a pair of brackets ([ ]) on either side of the field.

C. Press <RETURN> when you are finished typing, to confirm what you have entered. The menu erases the brackets to indicate that the information has been received. (Remember that as long as the brackets are displayed, the menu is waiting for information.)

D. Press <TAB> to move the typing cursor to the next field in the pop–up menu.

E. When you have finished entering information, slide the locator cursor off the pop–up menu, and it will disappear.

4. Select "Start test."

The test runs and reports the result of each check, including any errors that are encountered, in the transcript pad. Figure 4–4 shows the results of a successful loopback test.

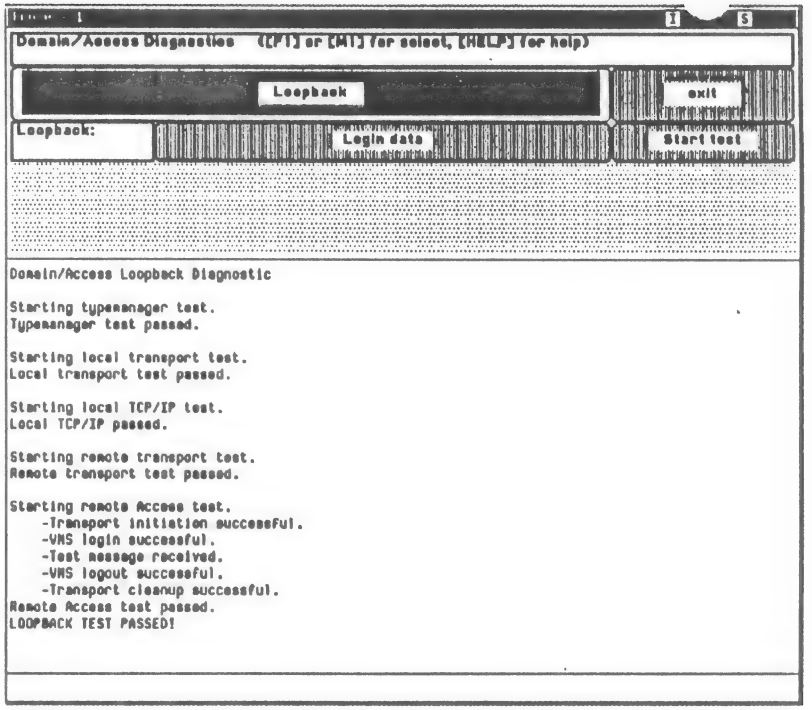

Figure 4–4. A Successful Loopback Test

Correcting Errors

If you enter a value in the pop–up menu that is not valid or you do not supply a value that is required, the menu displays a diagonal slash across the field and a smaller pop–up menu containing an error message appears in the field. For example, Figure 4–5 shows the pop–up that appears if you select "Start test" for the loopback test before you enter the log–in data.

To correct the error, move the cursor off the message pop–up, and the messages disappear. Then move the cursor to each empty or invalid field and enter the required value. Then, reselect the option that resulted in the error display.

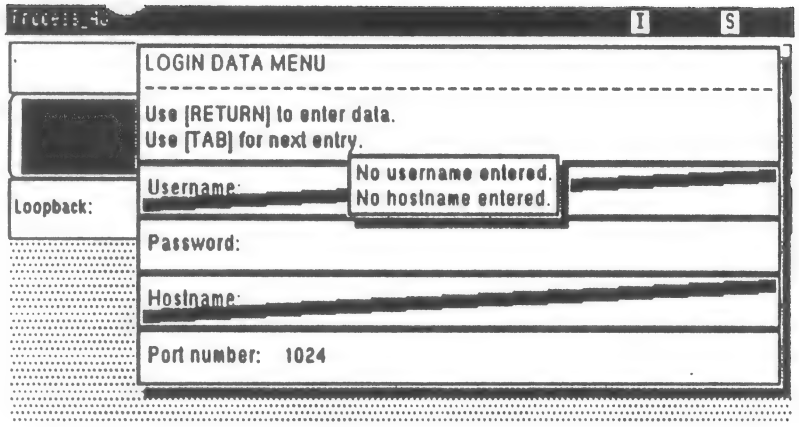

Figure 4–5. A Pop–up Menu that Indicates an Error

Requesting Help

To get help about a particular item in the menu, first point at (move the cursor over) the item; then press the HELP key. The help text pops up. If you move the locator cursor out of the help text pop–up, the help text disappears. Figure 4–6 shows the help text for the login data pop–up menu.

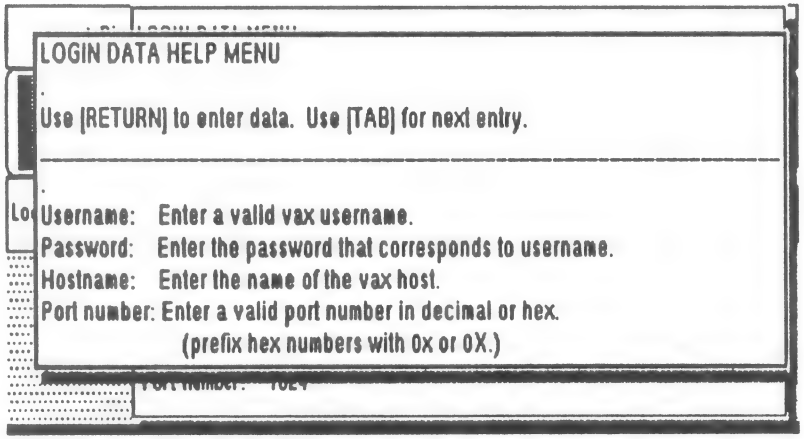
LOGIN DATA HELP MENU
Use [RETURN] to enter data. Use [TAB] for next entry.
Username: Enter a valid vax username.
Password: Enter the password that corresponds to username.
Hostname: Enter the name of the vax host.
Port number: Enter a valid port number in decimal or hex.
(prefix hex numbers with 0x or 0X.)

Figure 4–6. The Help Menu for the Login Data Pop–up Menu

Interpreting Error Messages

The following paragraphs list several error messages that can appear during the loopback test, indicate possible causes, and describe corrective actions.

Error Messages for the Local TCP/IP Test

The following messages can appear following the "Starting local TCP/IP test" message.

`TCP/IP server is not running (TCP/IP/library)`
The **tcp_server** process is not running; restart it.

`requested address not available (socket manager/manager)`
The TCP/IP port number is set to 0. Select login data and change the port number, then rerun the loopback test.

Error Messages for the Remote Transport Test

The following messages can appear following the "Starting remote transport test" message.

`object not found even though name exists (stream manager/IOS)`
TCP/IP at your workstation could not find the host name you specified in its configuration table. Check the **/sys/tcp/hostmap/local.txt** file for the VAX host name. If it is not there, follow the procedure for adding hosts that is described in the Domain TCP/IP documentation.

`connect timeout (access/apollo presentation)`
This message can have any of several causes, as follows:

- TCP/IP is not running on the VAX. Restart TCP/IP on the VAX.
- The VAX failed to recognize your host's name. Check the VAX TCP/IP host table and add your host to it if it is missing.

- There is no Domain/Access server process on the VAX with the port number you specified. Check to make sure that the port number you specify corresponds to the port number of the Domain/Access server running on the VAX. (You specify the port number when you run the @access command file on the VAX; for more information, see Chapter 2.) Start the required server process on the VAX or select "login data," specify the correct port number, and rerun the test.

Error Messages for the Remote Access Test

The following message can appear following the "Starting remote access test" and "Transport initiation successful" messages.

`login failed (VMS) (access/vms presentation)`
The user ID or password were not accepted by the VAX. You may have entered incorrect information, or the account may not be allowed to login into the VAX at this time. Verify the information and try again. If it still fails to login, attempt to login with a telnet window to verify that the account is still valid.

Appendix A

Planning Checklist

Use this appendix along with Chapter 1 to plan your software, hardware, and telecommunications facilities for the installation of the Domain/Access product.

Does Your Site Meet These Requirements?

☐ **Domain software properly installed and running**

- Aegis *or* Domain/IX operating system (SR 9.5 or later)

- *One* of the following Apollo TCP/IP products:
  - — Domain TCP/IP (version 3.0 or later)
  - — Domain/IX-BSD4.2 TCP/IP (version 3.0 or later)

☐ **VAX software properly installed and running***

- VMS operating system, Version 4.2 or later *or* MicroVMS operating system, Version 4.2 or later
- If using the VMS operating system, one of the following TCP/IP software packages:

  EXOS 8043 (version 3.2 or 3.3) TCP/IP Protocol Package for VAX/VMS Systems, by Excelan, Inc.

  or

  WIN/VX (version 2.3), or WIN/TCP (version 3.0), by The Wollongong Group, Inc.
- If using the MicroVMS operating system, one of the following TCP/IP software packages:

  EXOS 8044-01(version 3.2 or 3.3) TCP/IP Protocol Package for MicroVAX II Systems, by Excelan, Inc.

  or

  WIN/μVX (version 2.3), by The Wollongong Group, Inc.

☐ **Apollo network properly cabled and installed**

☐ **Domain TCP/IP node properly cabled and installed**

☐ **VAX computer properly cabled and installed***

** For a complete list of the latest supported configurations, see your sales representative.*

☐ **ETHERNET controller for VAX computer properly cabled and installed***

- If using Excelan software, choose one of the following:

  EXOS 204 ETHERNET Front-End Processor for UNIBUS Systems, by Excelan, Inc.

  or

  EXOS 203 ETHERNET Front-End Processor for Q-bus Systems, by Excelan, Inc.
- If using Wollongong software, choose one of the following:

  Digital Equipment UNIBUS Network Adapter (DELUA or DEUNA) by DEC

  or

  ETHERNET controller for UNIBUS systems, by Interlan, Inc.

  or

  Digital Equipment Q-bus Network Adapter (DEQNA), by DEC

☐ **ETHERNET cable properly installed**

☐ **ETHERNET transceivers properly cabled and installed**

** For a complete list of the latest supported configurations, see your sales representative.*

Appendix B

Command Reference

This appendix includes full descriptions of the VAX/VMS command file **@access** and the following shell commands:

- **crvg** (Create VAX/VMS Gateway Object)
- **dlvg** (Delete VAX/VMS Gateway Object)
- **svg** (Show VAX/VMS Gateway Object)

The **@access**, **crvg**, **dlvg**, and **svg** commands are supplied with the Domain/Access application.

@access — Start a Domain/Access Server on the VAX

FORMAT

@access *[port] [directory] [debug_level] [process_name] [tcp_type]*

The **access** command file is a VAX/VMS file that must reside in the **sys$manager** directory of the VAX node that is the TCP/IP host. This command file initializes the Domain/Access file server that runs on the VAX/VMS system to support file access requests from Apollo nodes.

You execute the **access** command file from the **sys$manager** directory on the VAX/VMS system. Therefore, you must log on to the VAX system and run this file. To load the **access** command file on the the VAX/VMS system, follow the configuration procedures in Chapter 2.

ARGUMENTS

port
(optional)

The TCP port number used by Domain/Access file server. This argument must be a value greater than 1024.

Default if omitted: The Domain/Access server uses TCP port 1024.

directory
(optional)

The VAX/VMS directory where the server resides.

Default if omitted: The **sys$system** directory.

debug_level
(optional)

The type of error messages to be displayed in your telnet window. Specify * for fatal errors on the console (displayed on the VAX console instead of in your telnet window), **f** for fatal errors in telnet window, **p** for presentation layer messages, **y** for application layer messages, **d** for data dumps, **i** for RMS error messages, **n** for presentation layer and application layer messages (combines **p** and **y**), or **0** (zero) for no errors at all.

Default if omitted: *, for the display of fatal errors on the VAX console.

process_name
(optional)

The name of the access server process on the VAX.

Default if omitted: **v1**_*port*, where port is the TCP port number for the Access server.

tcp_type
(optional)

The name of the TCP/IP software on the VAX computer. Specify **e** for Excelan or **w** for Wollongong.

Default if omitted: **e**, for Excelan software.

EXAMPLE

Start the Domain/Access file server on a VAX/VMS system, using TCP port 2100. You are logged on to the VAX system, either locally or by running telnet at your Apollo workstation.

```
@access 2100
```

crvg — Create a VMS Gateway Object

FORMAT

crvg *gate_name host_name user_name password [vax_node] [-p port_no] [-r] [-a account]*

The **crvg** command creates a Domain VMS gateway object (of type **vl_gate**) that contains information about the VAX/VMS system that will be accessed by Domain/Access users.

Once you execute this command, any filename that logically includes the name of the gateway object automatically refers to a VAX/VMS file or directory. The part of the pathname that is to the left of the gateway object name must conform to Domain file-naming conventions, and the portion to the right of the gateway object name must represent a VAX/VMS file or directory.

ARGUMENTS

gate_name (required)
: The name of the VMS gateway object that contains the information provided by the **host_name** and **vax_node** arguments. Once you execute the **crvg** command, using this name in a file pathname signals that you are accessing a VAX/VMS file.

host_name (required)
: The TCP/IP communications protocol host name identifier for the VAX with which your node communicates when a file pathname includes the gateway object name. Make sure that you supply a *case sensitive* name.

  If multiple VAX systems are connected using DECnet, the VAX that is the TCP/IP host does not have to be the target VAX that contains the files to be accessed.

user_name (required)
: The user_name argument specifies the name to use for logging in to the VAX. This argument is case-insensitive.

password (required)
: The password argument specifies the password to use for logging in to the VAX. This argument is case-insensitive.

vax_node (optional)
: The node name of the VAX/VMS system that contains the files to be accessed. *Do not* specify this argument if the VAX node that is the TCP/IP host and the VAX node that you wish to access are the same node.

  Default if omitted: Access files on the VAX/VMS system that is the TCP/IP host. This is the equivalent to specifying the null node name, "::".

OPTIONS

-p *port_no*
: Use the specified TCP port number for communications. If you specify a non-default port number, you must also specify this port number when you configure the Domain/Access software on the VAX/VMS system.

  All VMS gateway objects that will be used to communicate with a particular VAX/VMS TCP host (specified in the **host_name** argument) must use the same port number.

  This option allows you to specify an unused port number if the default port is already assigned to other software that runs on an Apollo node or VAX/VMS system.

  Default if omitted: Use TCP port 1024.

-r
: Replace any existing file system objects that have the same name with this gateway object.

-a This option allows you to specify a VAX account name for the login sequence. This is not required and is used for accounting purposes on the VAX.

The **crvg** command uses the command line parser, and so also accepts the standard command options (-help, -usage, -version) that are described in the *Domain System Command Reference*.

EXAMPLE

Create a v1_gate object named *v_paris* in the *//node_a* node directory. The VAX user_name is 'smith' which has a password 'nafriel'. Files accessed using this VMS gateway object are on the VAX/VMS node *paris::*. The VAX/VMS host that's used for TCP/IP communications is a system with the Internet name *vax1*. Communications will use TCP port *1024*, the Domain/Access default port.

```
crvg //node_a/v_paris vax1 smith nafriel paris::
```


dlvg — Delete a VMS Gateway Object

FORMAT

dlvg *gate_name*

The **dlvg** command deletes a Domain/Access VMS gateway object (object type **v1_gate**). Once you execute this command, you cannot use the deleted gateway object in any pathnames.

ARGUMENTS

gate_name (required) The name of the gateway object that you are deleting. This must be a gateway object name that was specified in a **crvg** command.

This argument accepts Domain wildcards. When you use wildcards, the command prompts you for a yes (y) or no (n) response before deleting each VMS gateway object.

The **dlvg** command uses the command line parser, and so also accepts the standard command options (-help, -usage, -version) that are described in the *Domain System Command Reference*.

EXAMPLE

Delete the VMS gateway object named *moscow* from the *//node_a* node directory.

```
dlvg //node_a/moscow
```

svg — Show Parameters for VMS Gateway Objects

FORMAT

svg *[gateway_name]*

The **svg** command displays the Internet host name, TCP port number, and VAX/VMS node ID specified by a Domain/Access VMS gateway object (object type **vl_gate**).

ARGUMENTS

gateway_name (optional)

The name of the gateway object for which you want information. This must be a VMS gateway object name that was specified in a **crvg** command.

This argument accepts Domain wildcards.

Default if omitted: Show parameters for all VMS gateway objects in your current directory.

The **svg** command uses the command line parser, and so also accepts the standard command options (–help, –usage, –version) that are described in the *Domain System Command Reference*.

EXAMPLE

Show the host names and node names that correspond to all VMS gateway objects that start with "v_" in *//node_a*'s root directory.

svg //node_a/v_?*

```
gateway name        tcp host name    user_name    password
//node_a/v_paris    vax1             smith        nafriel
//node_a/v_london   vax1             smith        thecat

account         port        gateway point to
research        1024        paris::
research        1024        london::
```

Glossary

This glossary contains definitions from *The Digital Dictionary, A Guide to Digital Equipment Corporation's Technical Terminology*, Second Edition, Copyright 1986. These definitions are indicated by an asterisk (*).

@access A VAX/VMS command file that starts the Domain/Access server on the VAX node that is the TCP/IP host.

account* A key to the system and a unit of system accounting. Each system user, including parts of the system itself, has an account. The system manager creates these accounts in the master file directory and assigns an account number to each. Individual users are identified by an account name as well. When you log in, you log in under a particular account name or number. This number informs the system where your files are and what kind of access to other files and system facilities you should be given.

account name* A string that identifies a particular account used to accumulate data on a job's resource use. This name is the user's accounting charge number, not the user's user identification code (UIC).

ACP* Ancillary control process.

ancillary control process (ACP)* An ancillary control process is an interface operation between user software and an I/O driver. An ACP provides functions supplementary to those performed in the driver, such as file and directory management.

AST*

Asynchronous system trap.

asynchronous system trap (AST)*

A software–simulated interrupt to a user–defined service routine. ASTs enable a user process to be notified asynchronously with respect to its execution of the occurrence of a specific event. If a user process has defined an AST routine for an event, the system interrupts the process and executes the AST routine when that event occurs. When the AST routine exits, the system resumes the process at the point where it was interrupted.

command file*

A VAX/VMS file containing commands and data that the command interpreter can accept in lieu of the user's typing the commands individually on a terminal. Thus, command procedures provide a means of automatically passing commands to the operating system. In addition they permit users to employ such programming techniques as loops, counters, labels, and symbol substitution to set up elaborate command sequences that can be altered through user interaction. Command procedures can also be submitted to the system for processing as batch jobs.

console*

The manual control unit integrated into the VAX/VMS central processor. The console includes a serial line interface connected to a hard–copy terminal, which enables the operator to start and stop the system, monitor system operation, and run diagnostics.

DAP*

Data access protocol.

data access protocol (DAP)*

A set of standardized formats and procedures that facilitate the creation, deletion, transfer, and access of files between a user process and a file system in a network environment.

DECnet*

1. The DIGITAL software facility that enables a user to access information on a remote computer via telecommunications lines.

2. DIGITAL networking software that runs on nodes in both local and wide–area networks.

DEQNA*

Digital Equipment Q–bus network adapter.

DEUNA*

Digital Equipment UNIBUS network adapter.

Digital Equipment Q–bus Network Adapter (DEQNA)*

The adapter that converts MicroVAX and other Q–bus–based systems to the ETHERNET network.

Digital Equipment UNIBUS Network Adapter (DEUNA)*

The adapter that converts VAX and UNIBUS–based PDP–11 systems to the ETHERNET network.

Domain/Access application

An Apollo application that provides transparent file access to VAX/VMS computers.

ETHERNET network

A local area network that transmits 10 megabits per second on coaxial cable with CSMA/CD baseband signaling.

ftp

A file transfer protocol that is defined by the Defense Advanced Research Projects Agency (DARPA). Ftp uses TCP/IP as its underlying protocol.

gateway object A Domain object containing connection information that the Domain/Access type manager needs to establish a TCP/IP connection with the VAX.

gateway node A node containing a gateway server, which is a process that receives a packet from one network, translates it to the protocols of another network, and transmits it on that network.

indexed file* A file that contains records and a primary key index (and optionally one or more alternate key indices) used to process the records sequentially by index or randomly by index.

Internet Protocol (IP) A protocol defined by the Defense Advanced Research Projects Agency for internetwork communications. IP transmits blocks of data from sources to destinations.

IP Internet protocol.

QIO* Queue input/output.

queue input/output (QIO)* Queue I/O is the VAX/VMS system service that services $QIO and $QIOW requests. The Queue I/O Request system service prepares an I/O request for processing by the driver and performs device-independent preprocessing of the request. This system service also calls driver FDT (function decision table) routines.

quota* The total amount of a system resource, such as CPU time, that a job is allowed to use in an accounting period, as specified by the system manager in the user authorization file.

record management services (RMS)* The more sophisticated of two sets of routines (supplied on RSX–11M–PLUS Systems) that are used to open and close files, read from files, write to files, and extend and delete files. RMS supports three forms of file organization and three forms of file access.

RMS* Record management services.

subserver A process on the VAX computer that is created by the Domain/Access server to manage the execution of commands and system calls for a particular user.

TCP Transmission control protocol.

TCP/IP Transmission Control Protocol/Internet Protocol. An acronym used to refer to the TCP and IP protocols and related Internet protocols, such as ftp and telnet, that are defined by the Defense Advanced Research Projects Agency.

telnet A remote terminal emulation protocol defined by the Defense Advanced Research Projects Agency (DARPA) for internetwork communications. Telnet uses TCP/IP as its underlying protocol.

timer* Two system processes: one that maintains the time of day and the date, and another that scans for device timeouts and performs time–dependent scheduling upon request. The time interrupt service routine creates the timer process.

transmission control protocol (TCP) A connection-oriented protocol defined by the Defense Advanced Research Projects Agency for internetwork communications. TCP provides for reliable communications between two processes in host computers, and uses the Internet Protocol as its underlying mechanism.

type manager A Domain object that translates Domain object types and record formats into those of a foreign file system, and vice versa.

UNIBUS* An asynchronous information bus for the PDP-11 universal basic utilities system.

user authorization file* A file containing an entry for every user that the system manager authorizes to gain access to the system. Each entry identifies the user name, password, default account, UIC (user identification code), quotas, limits, and privileges assigned to individuals who use the system.

VAX* Virtual address extension.

VAX/VMS* Virtual address extension/virtual memory system.

virtual address extension (VAX)* A computer made by the Digital Equipment Corporation. VAX is a high-performance, multiprogramming computer system based on a 32-bit architecture.

virtual address extension/virtual memory system (VAX/VMS)* VAX/VMS is the virtual memory operating system that runs on the entire range of VAX (32-bit) processors.

WIN Wollongong integrated networking.

working set* The set of pages in process space to which an executing process can refer without incurring a page fault. The working set must be resident in memory for the process to execute. The remaining pages of that process, if any, are either in memory and not in the process working set or they are on secondary storage.

Index

Primary page references are listed first. The letter *f* means "and the following page"; the letters *ff* mean "and the following pages". Symbols are listed at the beginning of the index. Task-oriented entries appear in color.

Symbols

[ ] (brackets) in **access_diag** pop-up menu, 4–4

@access B–2*f*

@access command file. *See* Domain/Access server

* (asterisk) as a debug level 2–9, 3–2*f*

/ (slash) in access_diag menu 4–5*f*

Numbers

0 debug level 2–9, 3–2

A

a option, for **crvg** command 2–16, B–6*ff*

access. *See* Domain/Access

access rights, required for VAX 2–2*ff*

access.com file 2–3*f*

access.hlp file 2–4*f*

access_diag diagnostic
- completing 4–3*ff*
- correcting errors with 4–5*f*
- error messages
  - from local TCP/IP test 4–7
  - from remote access test 4–8
- interpreting errors from 4–7*f*
- remote transport test error messages 4–7*f*
- requesting help with 4–6
- slash in menu 4–5
- starting 4–2
- using 4–1, 4–8

accessing files with DECnet 2–7, 2–13*ff*

account, showing 2–17

Aegis operating system, version required A–1*f*

application layer messages 2–9, 3–2*ff*

architecture of Domain/Access 1–3

arrow cursor, in **access_diag** 4–3

ast quota 2–7

astlm 2-

asynchronous system trap limit 2–7

authorize utility 2–6

B

batch jobs on VAX 3–9

byt quota 2–7

byte limit 2–7

bytlm 2–7

C

changing debug levels of the server 3–2*ff*

checklist, planning A–1*ff*

clipped windows 4–2

commands
- **@access** 2–8*ff*, B–2*f*
- @v1ssr 3–13*f*
- **crvg** 2–13*ff*, B–4*ff*
- **dlvg** 2–16, B–7
- **svg** 2–17, B–8*f*

completing the loopback test 4–3*f*

configuring the VAX computer 2–1*ff*

console, for VAX computer 2–8, 3–2

contacting customer services 3–3

controllers supported A–3

copying Domain/Access software to the VAX 2–2*ff*

correcting errors in **access_diag** 4–5*f*

creating
- gateway objects 2–12*ff*, B–4*ff*
- multiple TCP/IP connections 1–7
- subservers 1–5

crvg command 2–13*ff*, B–4*ff*

cursor locator, in **access_diag** 4–3

cursor, typing, in **access_diag** 4–4

customer services, contacting 3–3

D

d debug level 2–9, 3–2

data dumps 2–9, 3–2, 3–9

data type of image 2–5

debug level, for @v1ssr command 3–14

debug_level argument, for Domain/Access server 2–9, 3–2, B–3

debugging connections. *See* **access_diag** diagnostic

DECnet 2–13*f*
- showing 2–17

deleting gateway objects 2–16, B–7

DELUA controller A–3

detach privilege 2–7

DEQNA A–3

DEUNA A–3

diagnostic. *See* **access_diag** diagnostic

diagonal slash in **access_diag** menu 4–5

directory argument
- for **@v1ssr** command 3–14
- for Domain/Access server 2–9, B–2

dlvg command 2–16, B–7

DM process, 1–5
- assigning a name 2–10

Domain TCP/IP, version required A–1*f*

Domain TCP/IP product
- host tables 2–14
- port number 2–9, 2–15
- verifying communication 2–2

Domain/Access
- command syntax B–1*ff*
- commands
  - **@access** B–2*f*
  - @v1ssr 3–13*f*
  - **crvg** 2–13*ff*, B–4*ff*
  - **dlvg** 2–16, B–7
  - **svg** B–8*f*

Domain/Access application 1–1*ff*
- copying to the VAX 2–2*ff*
- introduction 1–2
- planning for 1–1*ff*

Domain/Access application architecture 1–3

Domain/Access Server 2–7
- debugging 3–1*ff*
- definition 1–4
- privileges to start 2–6*ff*
- starting the 2–8*ff*

Domain/IX operating system, version required A–1*f*

E

editing the startup file 2–10

error messages
- Excelan, Wollongong 3–3
- from **access_diag** 4–7*f*
  - local TCP/IP test 4–7
- from server 3–2*ff*
- getting more detail 3–2
- interpreting 3–8
- interpreting fatal errors 3–3
- remote access test, from **access_diag** 4–8
- remote transport test, from **access_diag** 4–7*f*

error messages from server 2–9

errors, correcting in **access_diag** menus 4–5*f*

errors in **access_diag** 4–7*f*

estimating load on VAX 1–8

estimating the number of users 1–8

ETHERNET, controllers A–3

ETHERNET network A–3

Excelan
- *See also* TCP/IP
- how to specify use 2–10
- **route add** command 2–11
- version required A–2

Excelan software
- how to specify use 3–14
- messages from 3–3

F

f debug level 2–9, 3–2*f*

fields in login data menu 4–3*f*

file protection, VAX/VMS 2–7

file size limitations 1–9

file types supported, VMS 1–2

format of error messages 3–3

ftp utility 2–2

G

gate_name argument
  for crvg command 2–14, B–4
  for dlvg command 2–16, B–7

gateway argument, for route add command 2–11

gateway objects
  creating 2–12*ff*, B–4*ff*
  deleting 2–16, B–7
  managing 2–12*ff*
  showing parameters of 2–17, B–8*f*

gateway objects, definition 1–4

gateway_name argument, for svg command 2–17, B–8

growing a window for access_diag 4–2

H

hardware, planning for 1–10, A–2*f*

hardware required A–2f

HELP key 4–6

help, VAX/VMS files, for Access 2–8

help menu, for access_diag 4–6

host tables
  Domain TCP/IP product 2–14
  TCP/IP 2–14

host_name argument, for crvg command 2–14, B–4

hostname, showing 2–17

I

i debug level 2–9, 3–2

i debug mode 3–9ff

image data type 2–5

indexed files, VMS 1–2

installing software 1–9, 2–1*ff*

Internet host name 2–17

interpreting
  application and presentation layer messages 3–8*f*
  RMS errors 3–9ff

interpreting data dumps 3–9

interpreting fatal errors 3–3*ff*

introduction to Domain/Access architecture 1–2

isolating problems 3–1*ff*

L

load on VAX, estimating 1–8

local TCP/IP test error messages from access_diag 4–7

locator cursor, in access_diag 4–3

logging in
  as the system manager of the VAX 2–2
  the user to the VAX 2–12

login data menu 4–3

login information, as part of gateway object 2–12

loopback test 4–3*f*

M

managing gateway objects 2–12*ff*

matching
  host names 2–14
  log-in information 2–8
  name of VAX node 2–15
  port number 2–15

menu
  for errors 4–6
  for help in access_diag 4–6
  for login data 4–4
  for loopback test 4–3

menu for access_diag 4–2

messages
  See also error messages
  application layer 3–2
  application layer, interpreting 3–8
  from Excelan or Wollongong software 3–3
  from server 3–2*ff*
  presentation layer 3–2, 3–8

messages from server 2–9

MicroVMS, version required 1–10, A–2

N

n debug level 2–9, 3–2

netmbx privilege 2–7

network configuration for Domain/Access 1–4

number of users supported 1–8

O

operating system, requirements 1–9, A–1*f*

overwrite. *See* replacing file system objects

P

p debug level 2–9, 3–2, 3–8

p option, for crvg command 2–15, B–5

parameter file argument, for @v1ssr command 3–13

password
  error 3–8
  showing 2–17

password argument, for crvg command 2–15, B–5

planing for hardware 1–10

planning
  for hardware A–2*f*
  for software A–2

planning checklist A–1*ff*

planning for software 1–9, A–1

pop–up menu, in access_diag 4–4, 4–6

port, error 3–4, 3–7

port argument B–2
  error 3–7
  for @v1ssr command 3–14
  for crvg command 2–15, B–5
  for Domain/Access server 2–9

port argument error 3–4

port, showing 2–17

preparing the VAX 2–6ff

presentation layer messages 2–9, 3–2, 3–8

privileges
  showing a process's 2–6
  showing someone else's 2–6

privileges for Domain/Access 2–6*ff*

problems, isolating 3–1*ff*

process_name argument, protection. *See* rights

process_name argument
  for @v1ssr command 3–14
  for Domain/Access server 2–10, B–3
  purging. *See* deleting gateway objects

Q

Q–bus controllers supported A–3

quotas for VAX 2–8

R

r option, for crvg command 2–15, B–5

reboot 2–10
  starting access after VAX reboot 2–10

remote access test, error messages from access_diag 4–8

remote transport test, error messages from access_diag 4–7*f*

removing. *See* deleting gateway objects

rename. *See* replacing file system objects

replacing file system object with same name as gateway object 2–15, B–5

requirements, for Domain/Access A–1*ff*

requirements for Domain/Access 1–3*ff*

response time 1–9

rights 2–7
  for copying Domain/Access to the VAX 2–2*ff*
  for using Access 2–7

RMS errors 2–9, 3–2, 3–9*ff*

running batch jobs on the VAX 3–9

S

server. *See* Domain/Access server

setprv privilege 2–7

shell process 1–5

showing
  account 2–17
  host name 2–17
  Internet host name 2–17
  parameters of gateway objects B–8*f*
  password 2–17
  RMS errors 3–9
  TCP port number 2–17
  user name 2–17
  VAX/VMS DECnet node ID 2–17

showing parameters, for gateway objects 2–17

showing type transformations 3–9

slash in access_diag menu 4–5

software
  access. *See* Domain/Access
  copying to the VAX 2–2*ff*
  EXOS. *See* Excelan software
  installing 2–1*ff*
  planning for A–1*f*
  version required A–2
  WIN. *See* Wollongong software

software installation 1–9

software planning 1–9*f*

spawning VAX processes 1–5

starting
  access_diag 4–2
  Domain/Access server B–2*f*

starting the Domain/Access server 2–8*ff*

starting the TCP/IP software on the VAX 2–11

status messages from server 2–9, 3–2*ff*

successful loopback test 4–5

svg command 2–17, B–8*f*

syntax, for Domain/Access commands B–1*ff*

sys$manager directory file 2–2*ff*

sys$manager:access.com file 2–7

sys$manager:v1ssr.com file 2–7

sys$startup.com file 2–10

sys$system directory file 2–2*ff*

sys$system:v1access.exe file 2–7

sys$system:v1server.exe file 2–7

sys$system:v1subserv.exe file 2–7

sysnam privilege 2–7

sysprv privilege 2–7

system manager, logging in as 2–2

T

TCP/IP
  connections 1–4*f*
  host tables 2–14
  local test error messages from access_diag 4–7
  port number 2–15, 2–17
  requirements, for Domain/Access A–1*ff*
  requirements for Domain/Access 1–3, 1–10
  supported on VMS 1–3
  verifying communications 2–2
TCP/IP port number 2–9
tcp_type argument, for Domain/Access server 2–10, B–3
tcphostname argument of route add command 2–11
telnet utility 2–2
telnet window, keeping open 3–2
test, loopback 4–3
timer queue limit 2–7
tmpmbx privilege 2–7
tqe quota 2–7
tqelm 2–7
transceivers for ETHERNET A–3
type
  gateway 2–12
  VMS file 1–2
type manager, treatment of gateway objects 2–12
type manager, definition 1–4
type transformation 3–9
typing cursor, in access_diag 4–4

U

understanding Domain/Access architecture 1–2*ff*
UNIBUS controllers supported A–3
UNIX operating system 1–10
user name, showing 2–17
user_name argument, for crvg command 2–15, B–5
username, error 3–8
users, number of 1–8
using, @v1ssr 3–13*f*
using access_diag 4–1*ff*

V

v1access.exe file 2–5*f*
v1server.exe file 2–5*f*
v1ssr.com file 2–3*f*
v1subserver.exe file 2–5*f*
varying the debug level of the server 3–2*ff*
VAX
  commands, show process/privilege 2–6
  file, size limitations 1–9
  files, systartup.com 2–10
vax_node argument, for crvg command 2–14, B–5*ff*
VAX/VMS
  commands, directory/protection 2–7
  computers supported 1–10
  configuring the 2–1*ff*
  console 3–2, 3–8
  copying software to the 2–2*ff*
  DECnet node ID 2–17
  file types supported 1–2
  files, help 2–8
  gateway objects
    creating 2–12*ff*, B–4*ff*
    definition 1–4
    deleting 2–16, B–7
    managing 2–12*ff*
    showing 2–17
    showing parameters of B–8*f*
  indexed files 1–2
  preparing the 2–6*ff*
  privileges to start 2–6ff
  processes 1–4*f*, 1–8
  quotas required 2–7
  sys$manager directory file 2–2*ff*
  sys$startup.com file 2–10*ff*
  sys$system directory file 2–2*ff*
  verifying TCP/IP communication 2–2
  version required 1–10
VMS
  authorize utility 2–6
  commands, show process/quota 2–8
  version required A–2
version
  error message 3–4
  required
    Aegis A–1*f*
    Domain TCP/IP product A–1*f*
    Domain/IX A–1*f*
    Excelan A–2
    MicroVMS A–2
    software A–2
    VMS A–2
  required Excelan 1–10
  required MicroVMS 1–10
  required Wollongong 1–10, A–2
  required VMS 1–10

W

WIN, Glossary–6
  See also Wollongong software
Wollongong
  how to specify use 2–10
  version required 1–10, A–2
Wollongong software
  See also TCP/IP
  how to specify use 3–14
  messages from 3–3
world privilege 2–7

Y

y debug level 2–9, 3–2, 3–8

Z

zero (0) debug level 2–9, 3–2